ALGORITMOS Y ESTRUCTURAS DE DATOS III - $1^{\underline{er}}$ Recuperatorio
Fecha examen: 11-DIC-2017 / Fecha notas: 15-DIC-2017

	Nº Orden	Apellido y nombre	L.U.	Cant. hojas[1]
Completar:				7
	Nota (Nº)	Nota (Letras)	Docente	
No completar:	8,6	OCHO		

1. Dados dos grafos $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$, se define su grafo junta como $G_1 + G_2 = (V_1 \cup V_2, E_1 \cup E_2 \cup V_1 \times V_2)$, es decir, el grafo que contiene a G_1 y a G_2 como subgrafos, y además contiene un eje entre cada vértice de G_1 y cada vértice de G_2.

   Sea J un grafo junta. Demostrar que J es conexo y J^c es no conexo. 2 p.

2. Sea G un grafo.
   (a) Demostrar que si G es autocomplementario entonces es conexo. 1 p.
   (b) Un vértice aislado de un grafo es un vértice que no es adyacente a ningún otro vértice del grafo. Un vértice universal de un grafo es un vértice que es adyacente a todos los otros vértices del grafo.
   Demostrar que si G es no trivial y autocomplementario entonces no tiene vértices aislados ni universales. 1 p.

3. Un grafo (simple) se dice 1-árbol si es un árbol con un eje agregado.

   Sea G un grafo de n vértices. Demostrar que son equivalentes: 2 p.
   (a) G es un 1-árbol.
   (b) G es conexo y tiene n ejes.
   (c) G es conexo y tiene un único ciclo.
   (d) G tiene n ejes y un único ciclo.

4. (a) Sea G un digrafo de $n \geq 2$ vértices y m arcos. Demostrar que si G *es* fuertemente conexo entonces $m \geq n$. Demostrar que si G *no es* fuertemente conexo entonces $m \leq (n-1)^2 = n(n-1) - (n-1)$. 1.25 p.
   SUGERENCIA: Para la segunda parte demostrar que a G le "faltan" al menos $n - 1$ arcos.
   (b) Exhibir una familia infinita de digrafos fuertemente conexos tal que el n-ésimo digrafo tenga n vértices y n arcos ($n \geq 2$). Justificar. 0.25 p.
   (c) Exhibir una familia infinita de digrafos no fuertemente conexos tal que el n-ésimo digrafo tenga n vértices y $(n-1)^2$ arcos ($n \geq 2$). Justificar. 0.25 p.
   (d) Se quiere demostrar la equivalencia de $n \geq 2$ predicados $p_1, p_2, \ldots p_n$. Para ello se planea demostrar implicaciones del tipo $p_i \Rightarrow p_j$. Mostrar que es necesario demostrar al menos n y a lo sumo $1 + (n-1)^2$ de dichas implicaciones. Indicar cuánto valen esas expresiones para el caso del Ejercicio 3. 0.25 p.

5. En una materia de la facultad están desarrollando un nuevo videojuego para enseñar programación dinámica. El videojuego es una adaptación del clásico Strikers 1945, que por algún motivo se va a llamar Dijkstrers 1930. Cada nivel del juego está dado por una cantidad de disparos disponibles y una matriz. El jugador maneja un avión propio que inicialmente se ubica en la posición inferior izquierda de la matriz. En cada una de las otras posiciones puede haber un avión enemigo. Cada segundo cada avión enemigo se mueve una fila hacia abajo, y el jugador puede elegir que el avión propio se mantenga quieto, o que simultáneamente se mueva una columna a la izquierda o a la derecha, siempre dentro de los límites de la matriz. Si el avión propio y un avión enemigo ocupan la misma posición, el jugador puede destruir al avión enemigo efectuando un disparo, aunque no está obligado a hacerlo. Si un avión enemigo llega a la fila inferior, en el paso siguiente desaparece del juego. Para completar el nivel, todos los aviones enemigos deben haber desaparecido del juego. El objetivo es completar el nivel habiendo destruído la máxima cantidad posible de aviones enemigos con los disparos disponibles. Diseñar un algoritmo eficiente que indique esa cantidad. La entrada del algoritmo es la cantidad $d \geq 1$ de disparos disponibles y la matriz de f filas y c columnas. El algoritmo debe tener complejidad $O(dfc)$. Mostrar que el algoritmo propuesto es correcto y determinar su complejidad. Justificar. El mejor algoritmo que conocemos tiene complejidad $O(fc)$, lo cual es necesario para obtener puntaje máximo en este ejercicio. En los siguientes ejemplos el avión propio se representa con $\triangle$ y cada avión enemigo con $\triangledown$. 2 p.

$d = 1$

$\triangledown$	$\triangledown$	$\triangledown$
$\triangledown$	$\triangledown$	$\triangledown$
$\triangle$	$\triangledown$	$\triangledown$

max = 1

$d = 314$

		$\triangledown$
	$\triangledown$	
$\triangle$		

max = 2

$d = 271$

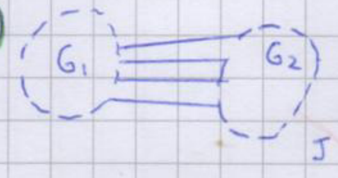
	$\triangledown$	
		$\triangledown$
$\triangle$		

max = 1

[1] Incluyendo a esta hoja. Entregar esta hoja junto al examen.

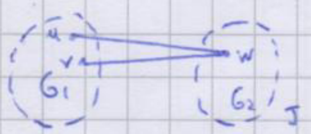
1) Sean los grafos $G_1 = (V_1, E_1)$ y $G_2 = (V_2, E_2)$, y el grafo junta $G_1 + G_2$ que contiene a G_1 y G_2 como subgrafos, y un eje entre cada vértice de G_1 y cada vértice de G_2, qvq si $J = G_1 + G_2 \Rightarrow J$ es conexo y J^c no lo es.

J)



Tomando $u, v \in V(J)$ vemos que tenemos los siguientes casos:

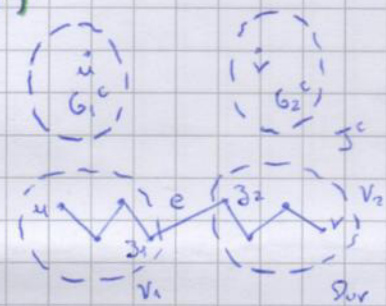
1. Si u y v pertenecen a distintos subgrafos (es decir, $u \in V_1 \wedge v \in V_2$ o viceversa) vemos que en J todo vértice de G_1 es adyacente a todo vértice de $G_2 \Rightarrow u$ y v son adyacentes y luego $\exists P_{uv}$ (camino entre u y v)



2. Si u y v pertenecen al mismo subgrafo (es decir, $u \in V_1 \wedge v \in V_1$ o viceversa) vemos que ambos son adyacentes a un vértice w en el subgrafo al que no pertenecen en la junta J. Luego $\exists P_{uw}$ y P_{vw} (caminos entre cada uno y w) y al juntarlos vemos que tenemos $P = P_{uw} + P_{vw} = P_{uv}$ camino entre u y v (notemos que los grafos no son orientados, por lo que $P_{vw} = P_{wv}$).

$\therefore$ $\forall u, v \in V(J)$ $\exists$ camino simple P_{uv} entre u y $v \Rightarrow J$ es conexo (por definición)

J^c)



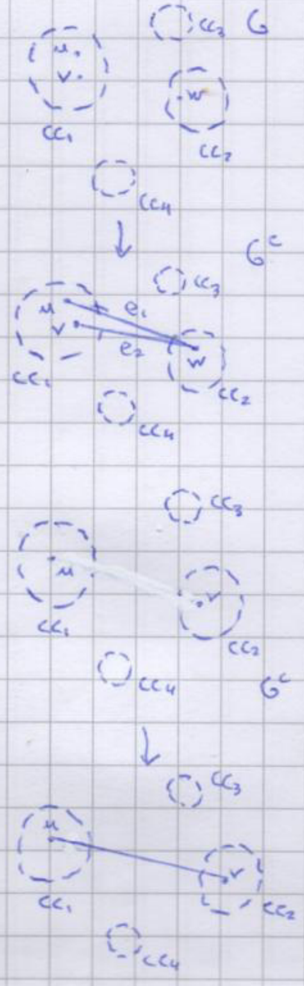
Tomemos $u \in V_1$ y $v \in V_2$. En la junta J sabemos que u y v son adyacentes pues por definición de junta $\forall u \in V_1, \forall v \in V_2\ \exists e = (u, v) \in X(J)$. Ahora en J^c u y v no son adyacentes (dicho eje no está presente).

Supongamos que $\exists P_{uv}$ en J^c. Viendo que $\forall v \in V(J^c)$, $v \in V_1 \vee v \in V_2$ llamamos z_1 al último nodo de P_{uv} en V_1 (yendo de u a v). Sabemos que z_1 existe pues $u \in V_1$ y $u \in P_{uv}$ (si no existiese luego todo nodo de P_{uv} sería de V_2 y luego $u \in V_2$ ABSURDO). Ahora, de z_1 parte un eje a un $z_2 \in V_2$ (pues tomamos el último nodo de V_1 en P_{uv} y existe el eje pues $v \in V_2$ y $v \in P_{uv}$). Llamándolo $e = (z_1, z_2)$ vemos que en J como $z_1 \in V_1$ y $z_2 \in V$ por la definición de junta $e' = (z_1, z_2) \in X(J)$. Luego en J^c, $e' = (z_1, z_2) \notin X(J^c)$ pero $e' = e$ ABSURDO (e no puede estar presente en J y J^c)

$\therefore$ $\exists u, v \in V(J^c)$ / $\nexists P_{uv}$ camino entre u y $v \Rightarrow J^c$ no es conexo (por definición)

2

2) a) Sea G un grafo autocomplementario (G y G^c son isomorfos) supongamos que G no es conexo.



Vimos por propiedad que si dos grafos G_1 y G_2 son isomorfos luego poseen la misma cantidad de componentes conexas $\Rightarrow$ G no es conexo $\Leftrightarrow$ G^c no es conexo. Sean $u, v \in V$ vemos que tenemos dos casos en G^c:

1- Si $u \in cc_i$ y $v \in cc_i$ en G (donde cc_i es la i-ésima componente conexa de G) vemos que en G^c ambos son adyacentes a $w \in cc_j$ donde $cc_j \neq cc_i$ (sabemos que son adyacentes a w pues si no lo fueran $w \in cc_i$ en G pero $cc_j \neq cc_i$, a su vez $\exists w \in cc_j \neq cc_i$ pues G no es conexo). Luego $e_1 = (u, w)$ y $e_2 = (w, v) \in X(G^c)$ y puedo armar el camino $P_{uv} = e_1 e_2 = u w v$ entre u y v en G^c

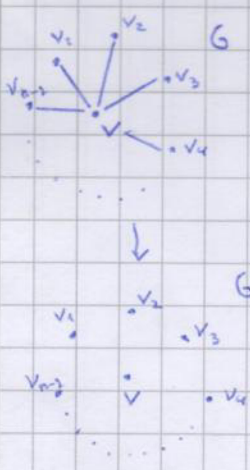
2- Si $u \in cc_i$ y $v \in cc_j$ donde $cc_i \neq cc_j$ ($\exists\, cc_i \neq cc_j$ pues G no es conexo) vemos que u y v no son adyacentes en G (si $e = (u, v) \in X(G) \Rightarrow u$ y v pertenecerían a la misma componente conexa de G).

En G^c, $e = (u, v) \in X(G^c) \Rightarrow u$ y v son adyacentes en G^c y $\exists P_{uv}$ camino entre u y v.

En ambos casos vemos que $\forall u, v \in V$ $\exists P_{uv}$ camino entre u y v en $G^c \Rightarrow G^c$ es conexo y por isomorfismo, G es conexo ABSURDO (supusimos que G no era conexo)

$\therefore$ G es autocomplementario $\Rightarrow$ G es conexo.

b) Sea G un grafo autocomplementario y $v \in V(G)$ q.v.q. $d(v) \neq 0 \wedge d(v) \neq n-1$ (siendo $n \geq 2$ por ser G no trivial)



G Si $d(v) = n-1$ luego $e = (v, u) \in X(G)$ $\forall u \in V \setminus \{v\}$, es decir, v es universal (adyacente a todo otro vértice de G).

En G^c $e = (v, u) \notin X(G)$ $\forall u\, V \setminus \{v\}$, es decir, v es un nodo aislado (no es adyacente a ningún otro vértice de G^c). Luego v pertenece a una cc (componente conexa) diferente del resto de los nodos de G^c.

G^c Sin embargo, aplicando lo visto en a) como G es autocomplementario $\Rightarrow$ G es conexo y como G es isomorfo a G^c $\Rightarrow$ G^c es conexo $\Rightarrow$ v no puede pertenecer a una componente conexa diferente del resto de los nodos de G^c ABSURDO

Si $d(v) = 0$ luego aplicando a) vemos que G es conexo y como $n \geq 2$ $\exists w \in V(G)$ / $(v, w) \in X(G) \Rightarrow d(v) \geq 1$ ABSURDO

$\therefore$ Si G es autocomplementario, $\forall v \in V$ $d(v) \neq n-1$ y $d(v) \neq 0$ (siendo G no trivial)

2

3) Debemos probar que siendo G un grafo de n vértices: (a) $\Leftrightarrow$ (b) $\Leftrightarrow$ (c) $\Leftrightarrow$ (d) donde:

(a): G es 1-árbol

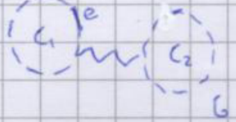
(b): G es conexo y tiene n ejes

(c): G es conexo y tiene un único ciclo

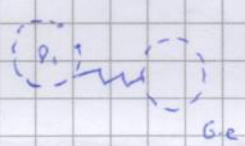
(d): G tiene n ejes y un único ciclo

(a) $\Rightarrow$ (b): Sea G un 1-árbol vemos que G es un árbol con un eje agregado. Por definición de árbol este es conexo $\Rightarrow$ al agregarle un eje sigue siendo conexo (si $G = A+e$ donde A es un árbol y e es un eje agregado, $\forall v \in G\ \exists P_{uv}$ en $A \Rightarrow \exists P_{uv}$ en $A+e$). A su vez, un árbol tiene $m = n-1$ ejes por lo que al agregarle 1 tiene $m = n-1+1 = n$ ejes. Luego G es 1-árbol $\Rightarrow$ G es conexo y tiene n ejes

(b) $\Rightarrow$ (c): Sea G un grafo conexo con $m=n$ ejes, supongamos que G posee más de un ciclo. Sabemos que $\exists C$ ciclo en G pues G es conexo y



$m > n-1$ (si no tuviese ciclos sería un árbol y tendría $m = n-1$ ejes al ser conexo). Ahora, de G tomamos los ciclos C_1 y C_2 / $C_1 \neq C_2$ (sabemos que existen pues supusimos que G tenía más de un ciclo). y vemos que como $C_1 \neq C_2 \Rightarrow$

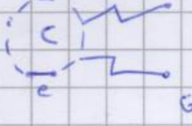


$\exists e / (e \in C_1 \wedge e \notin C_2) \vee (e \notin C_1 \wedge e \in C_2)$ (si $\forall e \in X$ se da que $e \in C_1 \Leftrightarrow e \in C_2 \Rightarrow C_1 = C_2$). Sin pérdida de generalidad tomamos $e \in C_1 / e \notin C_2$ y lo sacamos de G, obteniendo $G-e$

Sabemos por propiedad que como $e \in$ ciclo de $G \Rightarrow G-e$ es conexo y como G tiene n ejes $\Rightarrow G-e$ tiene $n-1$ eje $\Rightarrow$ $\Rightarrow G-e$ es árbol (conexo de $n-1$ ejes). Pero ahora, $\forall f \in C_2$ vemos que $f \in G$ pues $f \neq e$ ($e \notin C_2$) $\Rightarrow C_2 \in G-e \Rightarrow G-e$ tiene ciclos! ABSURDO

Luego, si G es conexo y tiene n ejes $\Rightarrow$ G es conexo y tiene un ciclo (G es conexo $\Rightarrow$ G es conexo es trivial)

(c) $\Rightarrow$ (d): Sea G un grafo conexo con un único ciclo, sabemos por propiedad que al ser conexo $m \geq n-1$. A su vez, como G posee un ciclo y es conexo, G no es árbol y $m > n-1$. Siendo $e \in C$ ciclo de G vemos que $G-e$ es conexo por propiedad siendo G conexo.



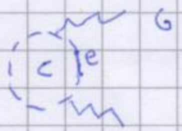
Como $e \in C$ y $e \notin G-e \Rightarrow C$ no está en $G-e$ y luego $G-e$ no posee ciclos. Si $\exists C'$ ciclo en $G-e \Rightarrow$ como $e \notin G-e$, $e \notin C' \Rightarrow C' \subseteq G$ y como $e \in C \wedge e \notin C' \Rightarrow C' \neq C$ ABSURDO (G tiene un único ciclo por hipótesis)

Luego $G-e$ es conexo y no tiene ciclos $\Rightarrow$ $G-e$ es árbol (por definición) $\Rightarrow$ $G-e$ tiene $n-1$ ejes.

Como $m(G-e) = m(G) - m(e) \Rightarrow n-1 = m(G) - 1 \Rightarrow m(G) = n$

$\therefore$ G es conexo y tiene un único ciclo $\Rightarrow$ G tiene n ejes y tiene un único ciclo (G tiene un único ciclo $\Rightarrow$ G tiene un único ciclo es trivial)

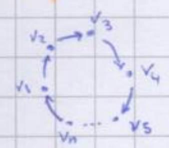
(d) $\Rightarrow$ (a): Sea G un grafo de n ejes y con un único ciclo, si G es 1-árbol $\Rightarrow$ G es un árbol con un eje agregado. Luego, tomamos



$e \in C$ ciclo único de G y formamos el grafo $G-e$. Como $e \in C$ y C es el único ciclo de $G \Rightarrow G-e$ no tiene ciclos (si $\exists C'$ ciclo de $G-e \Rightarrow e \notin C' \Rightarrow C' \subseteq G$ y como $e \notin C'$, $C = C' \Rightarrow G$ no tiene un único ciclo). A su vez, como G tiene n ejes, al sacarle 1 tiene $n-1$ ejes. Luego $G-e$ no tiene ciclos y tiene $n-1$ ejes $\Rightarrow$ $G-e$ es árbol (por definición) y vemos que G se forma agregando e a $G-e \Rightarrow G$ es 1-árbol

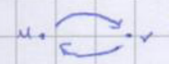
Habiendo probado que (a) $\Rightarrow$ (b) $\Rightarrow$ (c) $\Rightarrow$ (d) $\Rightarrow$ (a) por transitividad vale que (a) $\Leftrightarrow$ (b) $\Leftrightarrow$ (c) $\Leftrightarrow$ (d)

4) b) Sea la familia de digrafos $G_n = C_n$ donde C_n es un ciclo orientado de n nodos, vemos que $\forall n \geq 2$ C_n es fuertemente conexo pues $\forall u,v \in V(G_n)$



$\exists P_{uv}$ camino orientado de u a v: si $e=(u,v) \in C_n \Rightarrow P_{uv} = e$. Si $e=(u,v) \notin C_n \Rightarrow$ como $C_n = u\, z_1, z_2 \cdots z_k\, v\, z_{k+1} \cdots z_{n-2}$

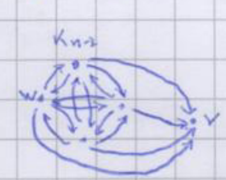
luego $P_{uv} = u\, z_1, z_2 \cdots z_k\, v$. Por ser C_n un ciclo orientado simple de n nodos $\Rightarrow C_n$ tiene n arcos



Para $n=2$ tenemos los nodos u y v adyacentes entre sí y vemos que $\exists P_{uv} = (u,v)$ y $P_{vu} = (v,u)$. A su vez, este tiene 2 arcos

Luego $\forall n \geq 2$ $G_n = C_n$ es una familia de digrafos fuertemente conexos donde G_n tiene n arcos

c) Sea la familia de digrafos G_n = digrafo completo de $n-1$ nodos $(K_{n-1}) \cup v \,/\, d_{in}(v) = n-1 \wedge d_{out}(v) = 0$ vemos que como $d_{in}(v) = n-1$ entonces todo



nodo de K_{n-1} incide sobre v (G_n tiene n vértices y K_{n-1} tiene $n-1$). Ahora, por ser K_{n-1} un digrafo completo de $n-1$ vértices

todo nodo de K_{n-1} es predecesor de otro nodo de K_{n-1} ($d_{out}(v) \geq n-2 \;\forall v \in K_{n-1}$) y como a su vez es predecesor de $v \Rightarrow$

$\forall w \in K_{n-1}$, $d_{out}(w) = n-2$ (predecesor de los otros $n-2$ nodos de K_{n-1}) $+ 1 = n-1$. Siendo $d_{out}(v) = 0$ sabemos por propiedad que

$$m = \sum_{i=1}^{n} d_{out}(v_i) = \sum_{w \in K_{n-1}} d_{out}(w) + \underbrace{d_{out}(v)}_{0} \quad (V = \{v\} \cup V(K_{n-1})) = (n-1) + (n-1) + 0 = (n-1)^2 \Longrightarrow G_n \text{ tiene } (n-1)^2 \text{ arcos}$$

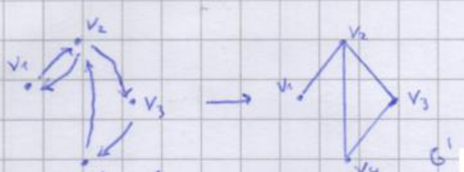
Ahora, vemos que G_n no es fuertemente conexo pues de v no sale ningún arco a algún otro nodo de G_n y sabemos que $\exists w \in V \,/\, w \neq v$ pues $n \geq 2 \Rightarrow$

$\nexists P_{vw}$ camino orientado de v a w en $G_n \Rightarrow G_n$ no es fuertemente conexo.

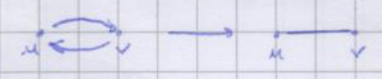
Para $n=2$ tenemos que $K_{n-1} = K_1 = u$ (digrafo trivial) y u incide sobre v pero v no incide sobre $u \Rightarrow \nexists P_{vu}$ camino orientado de v a u y G_2 no es fuertemente conexo. A su vez G_2 tiene 1 arco y $1 = (2-1)^2 = 1^2 = 1$

$\therefore$ $G_n = K_{n-1} \cup v \,/\, d_{in}(v) = n-1 \wedge d_{out}(v) = 0$ es una familia de digrafos no fuertemente conexos donde G_n tiene $(n-1)^2$ arcos $(n \geq 2)$

a) Sea G un digrafo de $n \geq 2$ vértices y m arcos, si G es fuertemente conexo, sabemos por propiedad que su grafo subyacente G' es conexo

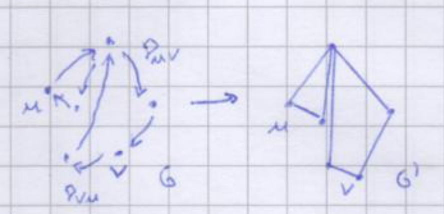


Como G' es conexo, sabemos por propiedad que $m(G') \geq n-1$ y como todo eje de G' proviene de al menos un eje orientado de $G \Rightarrow m \geq n-1$. Supongamos que $m = n-1$



Dados $u,v \in V$ vemos que en G no puede darse que estén los ejes (u,v) y (v,u) pues en G' estos forman un eje entre u y v. Al darse, la cantidad de ejes de G' es menor (o igual) que la de G y $m(G') < n-1$

pero G' es conexo $\Rightarrow m(G') \geq n-1 \Rightarrow$ si $e=(u,v) \in X(G) \Rightarrow e'(v,u) \notin X(G)$



Como todo par de nodos no tiene dos arcos opuestos entre sí, vemos que si $m = n-1 \Rightarrow m(G') = n-1$

y como G' es conexo $\Rightarrow G'$ es árbol (por definición).

Ahora sean dos nodos $u,v \in V$ vemos que como G es fuertemente conexo, $\exists P_{uv}$ y P_{vu} caminos orientados entre u y v. Como probamos que dos nodos z_1 y $z_2 \in V$ no pueden existir simultáneamente (z_1,z_2) y (z_2,z_1) vemos que P_{uv} y P_{vu} forman dos caminos entre u y v en G' diferentes (si fueran iguales luego P_{uv} y P_{vu} pasarían por los mismos nodos intermedios en orden inverso y tendríamos $z_1, z_2 \in P_{uv} \wedge P_{vu} \,/\, (z_1,z_2) \in P_{uv} \wedge (z_2,z_1) \in P_{vu}$). Con estos dos caminos sabemos por propiedad que en G' se forma un ciclo simple. Sin embargo G' es árbol ABSURDO (supusimos que G tenía $m = n-1$ arcos)

$\therefore$ Si G es fuertemente conexo $\Rightarrow G$ tiene $m \geq n$ arcos

5) Para calcular la cantidad de naves enemigas máxima que el avión del jugador puede destruir en una matriz de aviones enemigos al moverse como lo indica el enunciado, usaremos un algoritmo que se basa en la técnica de programación dinámica. Este tomará la matriz M por la que se debe mover el avión, su cantidad de filas f y columnas c, y la cantidad de disparos del avión al inicio d. Con esto formará una serie de subproblemas a través de la fórmula recursiva $F(i,j)$ donde i es la cantidad de pasos que pasaron y j es la columna en la que se encuentra el avión. Luego $F(i,j)$ = "máxima cantidad de aviones enemigos destruidos al pasar i pasos y estando el avión en la columna j".

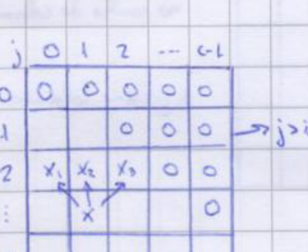
Para representar dichos subproblemas utilizaremos una matriz $S \in \mathbb{Z}^{c \times f}$ donde $S[i][j] = F(i,j)$. Vemos a su vez que $F(i,j)$ vale lo que sigue:

$$F(i,j) = \begin{cases} 0 & \text{si } i=0 \ \vee j > i \\ \min\{d, \max\{F(i-1,j-1), F(i-1,j), F(i-1,j+1)\} + M[f-1-i][j]\} & \text{si } i > 0 \wedge 0 < j < c-1 \end{cases}$$

(si $j=0$ tomamos este término sin $F(i-1,j-1)$ y si $j=c-1$ tomamos este término sin $F(i-1,j+1)$)

Con esto tenemos que, si suponemos que $M[i][j] = 1$ si hay un avión enemigo en la columna j y fila i de la matriz al comenzar el juego y $M[i][j] = 0$ si no lo hay, vemos que a medida que van pasando los pasos la cantidad de naves que el avión puede disparar es igual a la del paso anterior sumándole 1 si el avión se movió a una posición de una nave enemiga y tiene un disparo al menos disponible o 0 sino (en $F(i,j)$ vemos que el valor no puede superar d puesto que es la cantidad de disparos disponibles). Basándonos en la función recursiva F podemos describir el pseudo-código de nuestro algoritmo como sigue:

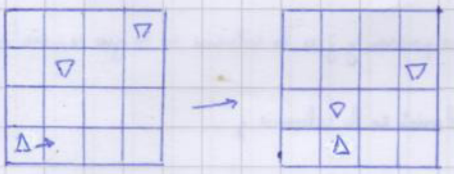
```
int maxMatados(M, c, F, d){
    S <- matriz de c columnas y F+1 filas inicializada en 0 en todas sus celdas
    maxCant <- 0

    para i de 1 a f {
        para j de 0 a min{i, c-1}{
            max <- S[i-1][j]
            si j>0 ∧ S[i-1][j-1] > max {
                max <- S[i-1][j-1]
            }
            si j<c-1 ∧ S[i-1][j+1] > max{
                max <- S[i-1][j+1]
            }
            max <- max + M[F-1-i][j]
            S[i][j] <- min(d, max)
        }
    }

    para j de 0 a c-1 {
        si S[F ][j] > maxCant {
            maxCant <- S[F ][j]
        }
    }

    devolver maxCant
}
```



$x = \min\{d, \max\{x_1, x_2, x_3\} + M[x]\}$ ($x = (i,j)$, $x_1 = (i-1, j-1)$, $x_2 = (i-1, j)$, $x_3 = (i-1, j+1)$)

Veamos ahora que el algoritmo devuelve el resultado correcto. Para ello, veamos primero que nuestra fórmula recursiva es correcta.

Sea $f(i,j)$ donde $0 \le i \le F$ y $0 \le j \le C-1$ vemos que i se define en dicho intervalo puesto que el juego comienza en el paso 0 ($i=0$) y el avión enemigo que desaparece por último es aquel que se encuentra en la fila mayor. Esto sale de ver que en ningún momento aparecen nuevos aviones en el juego y por cada paso decrecen una fila o desaparecen si no tienen fila inferior. Tras haber pasado F pasos todas las naves enemigas habrán desaparecido (como cada nave enemiga se encuentra en la fila i al inicio, $0 \le i \le F-1$ y por cada paso i decrementa en 1 hasta ser -1 y desaparecer la nave $\Rightarrow i^F = \max\{i-F, -1\} \le \max\{F-1-F, -1\} = -1$ donde i^k es la fila de la nave en el paso k)

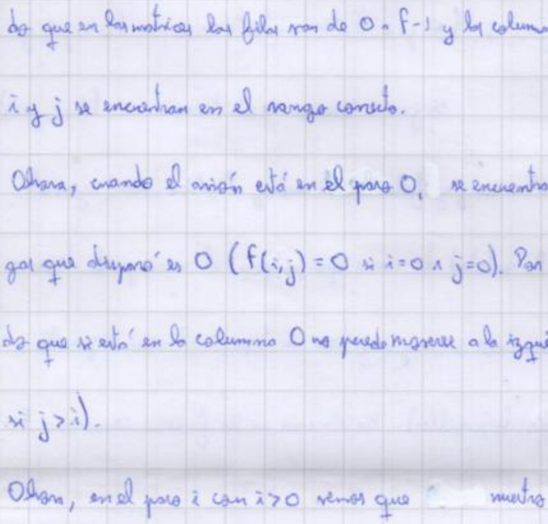


paso i paso $i+1$

Con esto vemos que $i \le F \Rightarrow 0 \le i \le F$. Por otro lado, el avión se encuentra en cada paso del juego en una columna j de la matriz y $0 \le j \le C-1$ (considerando que en las matrices las filas van de 0 a $F-1$ y las columnas de 0 a $C-1$ siendo F y C la cantidad de filas y columnas de la matriz respectivamente). Luego i y j se encuentran en el rango correcto.

Ahora, cuando el avión está en el paso 0, se encuentra en la esquina inferior derecha de M y no disparó a nadie, por lo que la cantidad de naves enemigas que disparó es 0 ($f(i,j)=0$ si $i=0 \wedge j=0$). Por otro lado, si pasaron i pasos, el avión pudo moverse como mucho i columnas a la derecha (suponiendo que si está en la columna 0 no puede moverse a la izquierda y pasar a la columna $C-1$) por lo que no puede estar en la columna j si $j > i$ ($f(i,j)=0$ si $j>i$).

Ahora, en el paso i con $i>0$ vemos que nuestro avión puede disparar a una nave enemiga si se encuentra en la misma posición que el avión. Como el avión se mantiene en la fila inferior y las naves enemigas descienden 1 fila x paso vemos que ambos se encuentran en la misma posición si la nave enemiga estaba inicialmente en la i-ésima fila (contando desde abajo, $F-1-i$-ésima contando desde arriba) y la j-ésima columna (la nave no cambia de columna al descender). Luego $M[F-1-i][j]$ indica que en la i-ésima fila y j-ésima columna había una nave enemiga que en el i-ésimo turno puede ser disparada.



paso 0   paso i

Siendo que por turno el avión dispara una nave o no más que en el turno anterior, y este puede moverse a izquierda, derecha o quedarse quieto en 1 columna, luego si queremos ver la mayor cantidad de naves disparadas en el turno anterior, vemos $\max\{f(i-1,j-1), f(i-1,j), f(i-1,j+1)\}$ (respetando los límites de la matriz). A esta cantidad le sumamos 1 si podemos disparar una nave enemiga en este turno, o 0 si no ($M[i][j]$) y vemos que como tenemos d disparos, el resultado no puede ser mayor a d ($f(i,j) = \min\{d, \max\{f(i-1,j-1), f(i-1,j), f(i-1,j+1)\} + M[i][j]\}$) puesto que no puede disparar a más de una a la vez.

Con esto vemos que $f(i,j)$ está bien definida y que obtenemos el resultado al problema como la máxima cantidad de naves disparadas pasados a lo sumo F pasos y estando el avión en alguna columna $0 \le j \le C-1$ (si el juego terminó antes luego en los siguientes pasos dicha cantidad no aumenta puesto que solo lo hace con $M[F-1-l][j]$ y $i > l$ que es la fila mayor de una nave enemiga, luego $M[F-1-l][j] = 0$), es decir $\max\limits_{0 \le j \le C-1}\{f(F,j)\}$

Ahora, en nuestro algoritmo comenzamos con la matriz S inicializada en 0 ($S[i][j]=0$ $\forall\, 0 \le i \le f$ y $\forall\, 0 \le j \le c-1$) y vemos que luego en un ciclo se recorre i de 0 a f ($0 \le i \le f$) y j de 0 a i ($j \le i$) tomando en máx el valor máximo de $S[i-1][k]$ con $k = j-1, j, j+1$, sumándole el valor de $M[f-1-i][j]$ (si puede destruirse la nave enemiga en el i-ésimo paso) y luego tomando el en caso de ser mayor este resultado (solo puede disparar al disparar). Esto hace que $S[i][j] = F(i,j)$ y notamos que por la forma de recorrer la matriz por cada paso tenga definidos los valores del paso anterior (i creciente).

Al finalizar el ciclo tenemos en $S[i][j]$ el valor de $F(i,j)$ $\forall\, 0 \le i \le f$ y $\forall\, 0 \le j \le c-1$ por lo que el resultado al problema lo devolvemos tomando el máximo $F(f,j)$ con $0 \le j \le c-1$ que es el máximo $S[f][j]$ con $0 \le j \le c-1$. Tomando de referencia el resultado de la función recursiva al problema, vemos que este es correcto.

faltó determinar la complejidad del algoritmo